Senium quick cheet:

### API workthough

1. Open a browser

# start an instance of firefox with selenium-webdriver

driver = Selenium::WebDriver.for :firefox

# :chrome -> chrome

# :ie -> iexplore

\* Go to a specified URL

driver.get 'http://google.com'

driver.navigate.to 'http://google.com'

`NOTE` -- the WebDriver may not wait for the page to load, you'd better using explicit and implicit waits.

\* Locating Elements

\* `find\_element` -- Find the first element matching the given arguments.

\* `find\_elements` -- Find all elements matching the given arguments

\* By ID

# example html

# <input id="q">...</input>

element = driver.find\_element(:id, "q")

\* By Class Name

# example html

# <div class="highlight-java" style="display: none; ">...</div>

element = driver.find\_element(:class, 'highlight-java')

# or

element = driver.find\_element(:class\_name, 'highlight-java')

\* By Tag Name

# example html

# <div class="highlight-java" style="display: none; ">...</div>

element = driver.find\_element(:tag\_name, 'div')

\* By Name

# example html

# <input id="q" name='search' type='text'>…</input>

element = driver.find\_element(:name, 'search')

\* By Link Text

# example html

# <a href="http://www.google.com/search?q=cheese">cheese</a>

element = driver.find\_element(:link, 'cheese')

# or

element = driver.find\_element(:link\_text, 'cheese')

\* By Partial Link Text

# example html

# <a href="http://www.google.com/search?q=cheese">search for cheese</a>

element = driver.find\_element(:partial\_link\_text, 'cheese')

\* By XPath

# example html

# <ul class="dropdown-menu">

# <li><a href="/login/form">Login</a></li>

# <li><a href="/logout">Logout</a></li>

# </ul>

element = driver.find\_element(:xpath, '//a[@href='/logout']')

\* `NOTE` -- When using Element#find\_element with `:xpath`, be aware that,

\* webdriver follows standard conventions: a search prefixed with "//" will search the entire document, not just the children of this current node.

\* Use ".//" to limit your search to the children of the receiving Element.

\* By CSS Selector

# example html

# <div id="food">

# <span class="dairy">milk</span>

# <span class="dairy aged">cheese</span>

# </div>

element = driver.find\_element(:css, #food span.dairy)

\* Element's operation

\* Button/Link/Image

driver.find\_element(:id, 'BUTTON\_ID).click

\* Text Filed

# input some text

driver.find\_element(:id, 'TextArea').send\_keys 'InputText'

# send keyboard actions, press `ctral+a` & `backspace`

driver.find\_element(:id, 'TextArea').send\_keys [:contol, 'a'], :backspace

\* Checkbox/Radio

# check if it is selected

driver.find\_element(:id, 'CheckBox').selected?

# select the element

driver.find\_element(:id, 'CheckBox').click

# deselect the element

driver.find\_element(:id, 'CheckBox').clear

\* Select

# get the select element

select = driver.find\_element(:tag\_name, "select")

# get all the options for this element

all\_options = select.find\_elements(:tag\_name, "option")

# select the options

all\_options.each do |option|

puts "Value is: " + option.attribute("value")

option.click

end

# anthoer way is using the Select class after seleniun-webdriver 2.14

element= driver.find\_element(:tag\_name,"select")

select=Selenium::WebDriver::Support::Select.new(element)

select.deselect\_all()

select.select\_by(:text, "Edam")

\* visibility

driver.find\_element(:id,'Element').displayed?

\* get text

driver.find\_element(:id,'Element').text

\* get attribue

driver.find\_element(:id, 'Element').attribute('class')

\* Driver's operation

\* execute javascript

driver.execute\_script("return window.location.pathname")

\* wait for a specific element to show up

# set the timeout to 10 seconds

wait = Selenium::WebDriver::Wait.new(:timeout => 10)

# wait 10 seconds until the element appear

wait.until { driver.find\_element(:id => "foo") }

\* implicit waits

An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available

driver = Selenium::WebDriver.for :firefox

# set the timeout for implicit waits as 10 seconds

driver.manage.timeouts.implicit\_wait = 10

driver.get "http://somedomain/url\_that\_delays\_loading"

element = driver.find\_element(:id => "some-dynamic-element")

\* switch between frames

# switch to a frame

driver.switch\_to.frame "some-frame" # name or id

driver.switch\_to.frame driver.find\_element(:id, 'some-frame') # frame element

# switch back to the main document

driver.switch\_to.default\_content

\* swich between windows

driver.window\_handles.each do |handle|

driver.switch\_to.window handle

end

\* handle javascript dialog

# get the alert

a = driver.switch\_to.alert

# operation on the alert

if a.text == 'A value you are looking for'

a.dismiss

else

a.accept

end

\* Cookies

\* Delete cookies

# You can delete cookies in 2 ways

# By name

driver.manage.delete\_cookie("CookieName")

# Or all of them

driver.manage.delete\_all\_cookies